Evaluation of the Transformer Architecture for Univariate Time Series Forecasting

Pedro Lara-Benítez[1], Luis Gallego-Ledesma[1], Manuel Carranza-García[1], and José M. Luna-Romera[1]

Division of Computer Science, University of Sevilla, ES-41012 Seville, Spain
plbenitez@us.es

Abstract. The attention-based Transformer architecture is earning increasing popularity for many machine learning tasks. In this study, we aim to explore the suitability of Transformers for time series forecasting, which is a crucial problem in different domains. We perform an extensive experimental study of the Transformer with different architecture and hyper-parameter configurations over 12 datasets with more than 50,000 time series. The forecasting accuracy and computational efficiency of Transformers are compared with state-of-the-art deep learning networks such as LSTM and CNN. The obtained results demonstrate that Transformers can outperform traditional recurrent or convolutional models due to their capacity to capture long-term dependencies, obtaining the most accurate forecasts in five out of twelve datasets. However, Transformers are generally more difficult to parametrize and show higher variability of results. In terms of efficiency, Transformer models proved to be less competitive in inference time and similar to the LSTM in training time.

Keywords: Time series · Forecasting · Attention · Transformers · Deep learning.

1 Introduction

Time series forecasting (TSF) is an important problem in machine learning with many practical applications in different domains such as energy demand [7], finance [19], or retail industries [5]. In recent years, deep learning (DL) models have become the most popular approach for TSF [20]. Architectures such as recurrent or convolutional networks have been specifically designed to deal with time series data, outperforming traditional statistical methods. DL models can automatically learn complex patterns without any prior assumptions on the data, achieving superior forecasting performance and being more scalable.

Long Short-Term Memory (LSTM) and convolutional (CNN) networks are among the most widely used architectures for TSF over the past years. More recently, Transformer models are gaining attention as a powerful alternative for time series processing. Unlike recurrent models, this architecture does not deal with the data in sequential order. Transformers can access any part of the history of the sequence using self-attention mechanisms, which makes them a potentially better solution to model long-term dependencies in the data.

This work presents an extension of the study conducted in [8], which provides the most extensive review of traditional deep learning techniques for TSF. In this study, we evaluate the performance of Transformer models under the same conditions and compare it with the best performing architectures that were LSTM and CNN. The experimental study uses 12 datasets with more than 50,000 time series from different fields to evaluate the forecasting precision and computational efficiency of the models. More than 200 architecture configurations of Transformer models are tested on each dataset, and the suitability of different hyperparameter choices is analyzed in-depth.

In summary, the main contributions of the study are the following:

- An extensive experimental study on Transformers models for univariate TSF.
- A comparative analysis with traditional state-of-the-art DL models.
- A thorough evaluation of different architecture and hyperparameter configurations of attention-based models for TSF.

The rest of the paper is organized as follows. Section 2 presents related studies. Section 3 describes the methodology and the materials used. Section 4 reports the experimental results. Section 5 presents the conclusions and future work.

2 Related Work

Deep learning architectures have become the most effective alternative for forecasting across related time series, since they allow building accurate global models that can learn shared features and dynamics. The study presented in [8] reviews the advantages and limitations of several DL models that have been proposed for TSF such as Long Short-Term Memory Networks (LSTM), Gated Recurrent Units (GRU), Echo State Networks (ESN), or Temporal Convolutional Networks (TCN). This study concludes that LSTM and CNNs provide the most robust performance across all the studied databases.

Very recently, attention-based models have also been applied to TSF with success. Some works have aimed to improve recurrent DL techniques using attention. For instance, in [3], an attention mechanism is used to enhance the selection of relevant timesteps in the past history for an encoder-decoder architecture using LSTMs. However, the Transformer architecture, which is purely based on self-attention mechanisms is recently earning more popularity. Transformers were first presented for machine translation in [21], showing since then an outstanding capacity to generalize to other tasks such as computer vision or sequence modeling.A self-attention model for capturing information across several dimensions (time, location, and measurements) was proposed in [14] for forecasting over geo-tagged time series. Later, an enhanced version of Transformer was presented in [10], which introduced causal convolution in the self-attention module in order to make the model more sensitive to the local context. Furthermore, they also provide some modifications to reduce the memory cost of Transformers, making it more feasible to deal with long time series. A novel Temporal Fusion Transformer was proposed in [12], combining recurrent and attention layers to

learn temporal dependencies at different scales over several real-world datasets. A deep Transformer model for influenza-like illness forecasting is presented in [23], which outperforms LSTM and Seq2Seq models. The self-attention of Transformers showed better forecasting performance than the linear attention used in Seq2Seq architectures.

3 Materials and Methods

This section describes the Transformer architecture for time series forecasting and the experimental setup. For reproducibility purposes, the complete implementation of the experiments is published at [9].

3.1 Attention-based Deep Neural Network

The Transformer is a deep learning architecture based on attention mechanisms. The Scaled Dot-Product Attention algorithm, introduced in [21], aimed to give the models the capacity to focus on the most relevant elements of long sequences. This is achieved by computing a weighted sum of the values (V), where the weights are computed applying the softmax function to the dot products of the queries (Q) with the keys (K), scaled by the square root of the dimension of the keys (d_k).

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

A variant of this algorithm, called Multi-Head attention, is used in the Transformer. This version applies h learnable linear projections to the queries, keys, and values before applying attention individually over each projection. Then, the results of each attention are concatenated before the last linear projection.

The original Transformer consisted of an encoder and a decoder. However, in this study, we consider a decoder-only architecture introduced in recent works as a more problem-agnostic model [10]. As can be seen in Figure 1, the Transformer consists of several stacked decoder blocks that pass the encoding from the previous decoder as the input to the following blocks.

Each decoder block is composed of a first masked self-attention layer followed by a multi-head attention layer and a feed-forward block. Furthermore, all the sub-layers use a residual connection followed by dropout and batch normalization layers, to improve the capacity of generalization of the network [13]. In addition, to model the sequential information of the time series, a positional encoded vector, generated with sine and cosine functions, is added to the input sequences.

The network is fed with a time series of fixed size ($forecast\_horizon$) and the target output is the same sequence right shifted. In order to prevent the model from paying attention to values in the future, a mask is used before the softmax function. The mask sets all upper triangular elements to $-\infty$ so that future information has no importance in the attention layer. Hence, the network will learn to predict the next value of the input sequence based only on

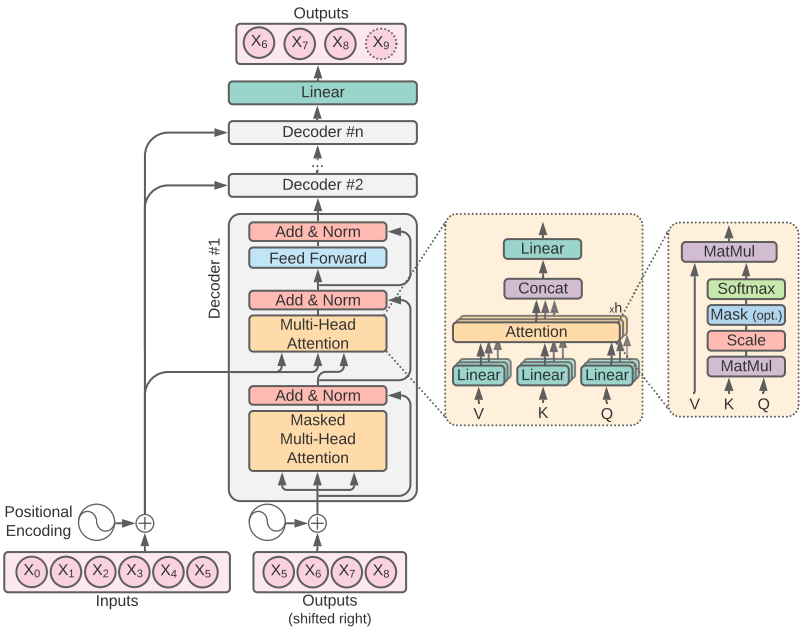

Fig. 1: Transformer architecture. In this example, the *past history* and *forecasting horizon* are 6 and 4 respectively.

the previous values. Afterwards, the multi-head attention is performed over the previous *past_history* elements of the sequence.

For multi-step-ahead forecasting problems, the inference is carried out by calling the model iteratively *forecast_horizon* times. Hence, the last prediction is included in the input to compute the next value. As this method propagates the error along the prediction sequence, to help model convergence during training, the teacher forcing scheme [22] is used, including the actual value at each new prediction.

3.2 Experimental Study

In this subsection, we present the design of the experimental study carried out to evaluate the Transformer architecture. The results obtained from the different architecture configurations over 12 datasets are analyzed and compared statistically with the state-of-the-art deep learning models.

3.2.1 Datasets. For the experimental study, we have used the same 12 publicly available datasets selected in [8], each of them with multiple related time series. These datasets, described in Table 1, present a wide diversity of characteristics in terms of length, domains, complexity, and seasonality.

3.2.2 Model Parametrization. This study aims to evaluate the performance of the Transformer model for TSF problems, in terms of both accuracy

Table 1: Description of datasets. Columns N, FH, M and m refer to number of time series, forecast horizon, maximum length and minimum length respectively.

#	Datasets	N	FH	M	m	Description	Ref.
1	CIF2016o12	57	12	108	48	Financial and artificially generated.	[24]
2	CIF2016o6	15	6	69	22		
3	ExchangeRate	8	6	7588	7588	Exchange rates of 8 countries.	[6]
4	M3	1428	18	126	48	Monthly time series of different domains.	[15]
5	M4	48000	18	2794	42		[16]
6	NN5	111	56	735	735	Daily ATMs cash withdrawals.	[17]
7	SolarEnergy	137	6	52560	52560	Solar power production records.	[18]
8	Tourism	336	24	309	67	Tourism data from Australia, Hong Kong, and New Zealand.	[1]
9	Traffic	862	24	17544	17544	Occupancy rates of California Department of Transportation.	[2]
10	Traffic-metr-la	207	12	34272	34272	Traffic speed of the highways of Los Angeles and the Bay area.	[11]
11	Traffic-perms-bay	325	12	52116	52116		
12	WikiWebTraffic	997	59	550	550	Web traffic of Wikipedia pages.	[4]

Table 2: Parameter grid search.

(a) Model architecture parameters.

Transformer		LSTM		CNN	
$\mathbf{d_{model}}$	16, 128, 256	**Units**	32, 64, 128	**Filters**	16, 32, 64
Layers	2, 3, 4	**Layers**	1, 2, 4	**Layers**	1, 2, 4
h	4, 8	**Return Seq**	True, False	**Pool size**	0, 2

(b) Training parameters.

Past history	1.25, 2, 3
Batch size	32, 64
Epochs	5
Optimizer	Adam
Learning rate	Same as [21]
Normalization	minmax, zscore

and efficiency. To this end, we have conducted an exhaustive grid search for both the architecture and training hyperparameters. As a result, a total of 216 Transformer models with different configurations have been trained and evaluated over each dataset. Furthermore, we compare the Transformer with the LSTM and CNN networks, as they achieved the best results in the previous study [8].

Table 2 presents the parameter search carried out for each architecture. For the Transformer, the dimension of the model (d_{model}), the number of stacked decoders ($layers$), and the number of linear projections in the multi-head attention (h) are fine-tuned. The possible values have been chosen based on what is commonly used in the literature, while also ensuring a fair comparison between architectures. Therefore, the same training hyper-parameters as in [8] are used, except for the learning rate, which is varied along the training process as indicated in the original study [21].

3.2.3 Evaluation procedure. For evaluating the models, the last part of each individual time series (forecast horizon) is used as the test set, while the

rest is used as training data. The same preprocessing steps as in [8] are applied, using the Multi-Input Multi-Output (MIMO) strategy to transform the time series into training instances that can feed the DL models.

This study analyses the best results obtained with each type of network, as well as the distribution of results of the different parameter configurations. The efficiency of the models is compared in terms of both training and inference time. The weighted absolute percentage error (WAPE) and the mean absolute error (MAE) metrics are used to measure the predictive performance:

$$WAPE = \frac{\frac{1}{n}\sum_{i=1}^{n}|y_i - o_i|}{\frac{1}{n}\sum_{i=1}^{n} y_i} \times 100\% \quad (2) \qquad MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - o_i|, \quad (3)$$

With the obtained results, a statistical analysis is carried out. Hommel's post-hoc analysis is conducted to find if there are significant differences between the performance of the models. Furthermore, we perform a paired Wilcoxon signed-rank test in order to study the statistical differences among the architecture configurations of each type of model.

4 Results and Discussion

This section presents the experimental results, which have been carried out using a computer with an NVIDIA RTX 2080 Ti 12GB GPU and an Intel i7-8700 CPU. An appendix with the full report of the results can be found at [9].

4.1 Forecasting accuracy

Table 3 reports the best WAPE and MAE results obtained by the Transformer, LSTM, and CNN networks for each time series dataset. Overall, the three architectures achieved similar results. Specifically, the LSTM lead the ranking 5 out of 12 datasets for both metrics, the Transformer achieves the top results 4 times, while the CNN wins only in 3 datasets for each metrics. It is important to mention that Transformers obtain the most accurate predictions in popular forecasting competitions such as M3 or M4, which are also the two largest datasets. On average, the LSTM obtains the first position in the ranking, closely followed by the Transformer. In fact, Hommel's post-hoc analysis carried out determines that there are no significant differences between the Transformer and the LSTM, while they are both significantly better than the CNN.

Figure 2 presents the distributions of WAPE obtained by each model for each dataset. In general, we can observe that the Transformer is more sensitive to the model parametrization than the other architectures, as it presents a wider distribution. In order to analyze the results obtained with the different architecture parameters and training hyper-parameters, we use the Wilcoxon statistical test. The results of this test are reported in Table 4, where ** indicates that it is the best value with a significant statistical difference compared to the rest ($p < 0.05$), the * indicates that there is a certain tendency suggesting that it is

better to choose that parameter ($p < 0.2$), and = means there are no significant differences between choosing any of the possible parameters.

Table 3: Best WAPE and MAE results obtained with each type of architecture for all datasets.

	Datasets	WAPE			MAE		
		Transformer	LSTM	CNN	**Transformer**	LSTM	CNN
1	CIF2016o12	**11.207**	12.475	12.479	12,564.18	**11,732.31**	12,762.73
2	CIF2016o6	16.157	**15.352**	17.143	**2,182,435.2**	3,636,929.7	2,833,131.5
3	ExchangeRate	0.303	**0.300**	0.335	0.0021	**0.0019**	0.0020
4	M3	**12.490**	15.282	15.612	**659.18**	700.25	709.44
5	M4	**13.587**	14.281	14.256	**588.38**	597.54	593.71
6	NN5	18.637	**18.589**	18.852	3.570	**3.538**	3.572
7	SolarEnergy	13.550	12.452	**11.717**	2.246	2.066	**1.977**
8	Tourism	18.68	19.081	**18.497**	2,202.11	2,280.09	**1,969.58**
9	Traffic	33.541	**31.960**	34.406	0.0121	**0.0114**	0.0124
10	Traffic-metr-la	3.418	3.359	**3.337**	2.029	2.009	**1.991**
11	Traffic-perms-bay	1.333	**1.314**	1.433	0.885	**0.870**	0.946
12	WikiWebTraffic	**46.110**	46.477	46.914	**11.796**	12.063	12.106
	Mean ranking	1.833	**1.750**	2.416	1.916	**1.833**	2.250

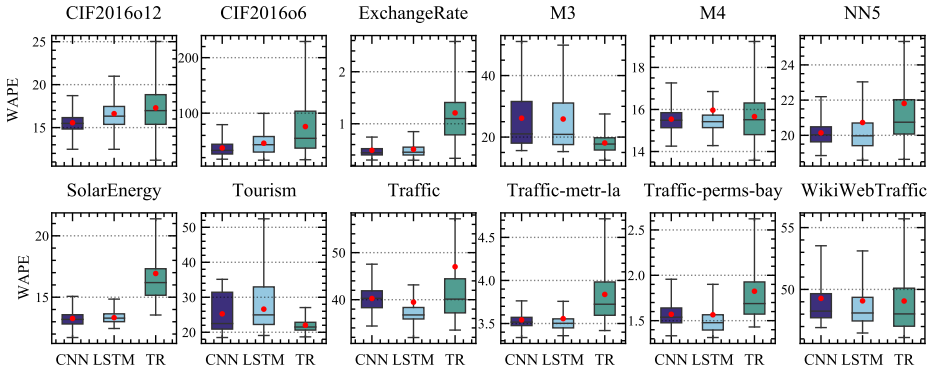

Fig. 2: Distribution of WAPE results obtained by each model architecture for each dataset.

Table 4: Architecture configuration analisys.

(a) Training hyper-parameters.

	LSTM	CNN	**TR**
Batch size	32**	64**	=
Past History factor	1.25**	1.25**	1.25**
Normalization Method	minmax**	zscore**	=

(b) Model architecture.

	TR
Layers	2**, 3*
$\mathbf{d_{model}}$	16**, 128*
h	4*

4.2 Computation time

We have also evaluated the different architectures in terms of computational efficiency. Figure 3 reports the distribution of training and inference time measured for each architecture. It is worth noting that the Transformer differs significantly from the other models in terms of inference time. This is due to the particularity of the Transformer architecture, which iterates generating single-step predictions for multi-step-ahead forecasting problems. Therefore, while CNN and LSTM compute a multiple steps prediction with a single call, the transformer will have to be called several times, specifically $forecasting\_horizon$ times. This behaviour is illustrated in Figure 4, where we can see how the inference time is directly proportional to the prediction horizon. More precisely, on average, the transformer generates a single prediction in 3.2 milliseconds. However, as it has to be called for each prediction horizon, the inference time increases to almost 200 when the horizon reaches 59 time steps.

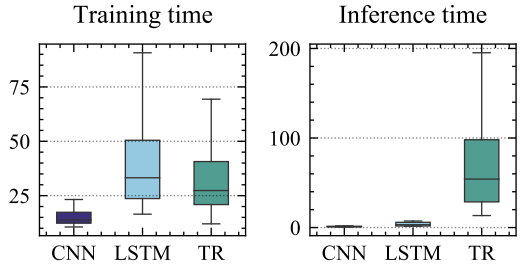

Fig. 3: Distribution of training and inference time by instance measured in milliseconds.

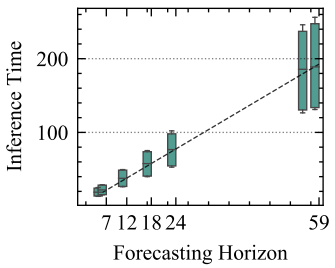

Fig. 4: Inference time of the transformer versus forecasting horizon.

In terms of training time, the Transformer is faster than the LSTM but not as fast as the CNN. The results show that in terms of the speed/accuracy trade-off, the CNN is the best one in terms of computation efficiency but with lower accuracy while, LSTM and Transformer behave similarly, achieving good forecasting accuracy but having a significantly slower training process compared to the CNN.

5 Conclusions

In this paper, we evaluated the performance of the Transformer architecture for time series forecasting in terms of accuracy and computational efficiency. An extensive experimental study over 12 datasets and different architecture configurations was carried out. The results are compared with long-short term memory (LSTM) and convolutional (CNN) networks, which are considered the state of the art in the field. The conclusion obtained from the analysis of the results of these experiments are summarized below:

- The Transformer architecture achieves state-of-the-art forecasting accuracy, obtaining similar results to the LSTM and outperforming CNNs.
- Transformers provide a better accuracy/speed trade-off than LSTM in training time. However, the Transformer training process is significantly slower compared to the CNN.
- The inference time of the Transformer architecture is severely influenced by the single-step prediction scheme used, which makes it slower than the other architectures.
- Finding the best architecture configuration for Transformers is a complex task as it presents a wider WAPE distribution than the other models.

In summary, the conclusions obtained from analyzing the results establish the Transformer architecture at the level of the state-of-the-art deep learning techniques for univariate time series forecasting. In future studies, alternative architecture variations such as convolutional attention or sparse attention should be considered. Another future study should work on non-auto-regressive models to reduce the inference time of the Transformers. Furthermore, we aim to study the use of multi-dimensional Transformers for dealing with spatio-temporal grid data.

Funding

This research has been funded by FEDER/Ministerio de Ciencia, Innovación y Universidades – Agencia Estatal de Investigación/Proyecto TIN2017-88209-C2 and by the Andalusian Regional Government under the projects: BIDASGRI: Big Data technologies for Smart Grids (US-1263341), Adaptive hybrid models to predict solar and wind renewable energy production (P18-RT-2778). We are grateful to NVIDIA for their GPU Grant Program that has provided us high-quality GPU devices for carrying out the study.

References

1. Athanasopoulos, G., Hyndman, R.J., Song, H., Wu, D.C.: Tourism forecasting part two. www.kaggle.com/c/tourism2 (2010)
2. Dua, D., Graff, C.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
3. Fan, C., Zhang, Y., Pan, Y., Li, X., Zhang, C., Yuan, R., Wu, D., Wang, W., Pei, J., Huang, H.: Multi-horizon time series forecasting with temporal attention learning. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 2527–2535. KDD '19 (2019). https://doi.org/10.1145/3292500.3330662
4. Google: Web traffic time series forecasting competition. www.kaggle.com/c/web-traffic-time-series-forecasting (2017)
5. Karmy, J., Maldonado, S.: Hierarchical time series forecasting via support vector regression in the european travel retail industry. Expert Systems with Applications **137**, 59–73 (2019). https://doi.org/10.1016/j.eswa.2019.06.060

6. Lai, G., Chang, W.C., Yang, Y., Liu, H.: Modeling long- and short-term temporal patterns with deep neural networks. arXiv:1703.07015 (2017)
7. Lara-Benítez, P., Carranza-García, M., Luna-Romera, J.M., Riquelme, J.C.: Temporal convolutional networks applied to energy-related time series forecasting. Applied Sciences **10**(7), 2322 (2020)
8. Lara-Benítez, P., Carranza-García, M., Riquelme, J.C.: An experimental review on deep learning architectures for time series forecasting. International Journal of Neural Systems **31**(03), 2130001 (2021). https://doi.org/10.1142/S0129065721300011, pMID: 33588711
9. Lara-Benítez, P., Gallego-Ledesma, L., Carranza-García, M.: Time Series Forecasting - Deep Learning. https://github.com/pedrolarben/TimeSeriesForecasting-DeepLearning (2021)
10. Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.X., Yan, X.: Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. arXiv:1907.00235 (2020)
11. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv:1707.01926 (2017)
12. Lim, B., Arik, S.O., Loeff, N., Pfister, T.: Temporal fusion transformers for interpretable multi-horizon time series forecasting. arXiv:1912.09363 (2020)
13. Luo, P., Wang, X., Shao, W., Peng, Z.: Towards understanding regularization in batch normalization. arXiv preprint arXiv:1809.00846 (2018)
14. Ma, J., Shou, Z., Zareian, A., Mansour, H., Vetro, A., Chang, S.F.: Cdsa: Cross-dimensional self-attention for multivariate, geo-tagged time series imputation. arXiv:1905.09904 (2019)
15. Makridakis, S., Hibon, M.: The M3-competition: results, conclusions and implications. International Journal of Forecasting **16**(4), 451 – 476 (2000). https://doi.org/10.1016/S0169-2070(00)00057-1
16. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: The M4 competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting **36**(1), 54 – 74 (2020). https://doi.org/10.1016/j.ijforecast.2019.04.014
17. NNGC: NN5 time series forecasting competition for neural networks. http://www.neural-forecasting-competition.com/NN5 (2008)
18. NREL: Solar power data for integration studies. www.nrel.gov/grid/solar-power-data.html (2007)
19. Sezer, O., Gudelek, M., Ozbayoglu, A.: Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. Applied Soft Computing Journal **90** (2020). https://doi.org/10.1016/j.asoc.2020.106181
20. Torres, J., Hadjout, D., Sebaa, A., Martínez-Álvarez, F., Troncoso, A.: Deep learning for time series forecasting: A survey. Big Data **9**(1), 3–21 (2021). https://doi.org/10.1089/big.2020.0159
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017)
22. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Computation **1**(2), 270–280 (1989). https://doi.org/10.1162/neco.1989.1.2.270
23. Wu, N., Green, B., Ben, X., O'Banion, S.: Deep transformer models for time series forecasting: The influenza prevalence case. arXiv:2001.08317 (2020)
24. Štěpnička, M., Burda, M.: Computational Intelligence in Forecasting (CIF). https://irafm.osu.cz/cif (2016)